SEEDS Reuse and Reference Architecture Study

Software Reuse Process Definition

Workshop 2 (June 17 – 19) Results

Breakouts with Community Input

Agenda

- **Workshop instructions (15 min)**
  - Workshop goals
  - Workshop format
  - Reuse goals
- **Group exercise (20 min)**
  - Problems & opportunities
- **Reuse process breakout teams (2 hr + 10 min break)**
  - Guiding principles
  - Reuse program strategies
  - Reuse enablement strategies
  - Evolutionary processes
  - Reference architecture purposes
  - Model reuse initiatives
  - Critical success factors

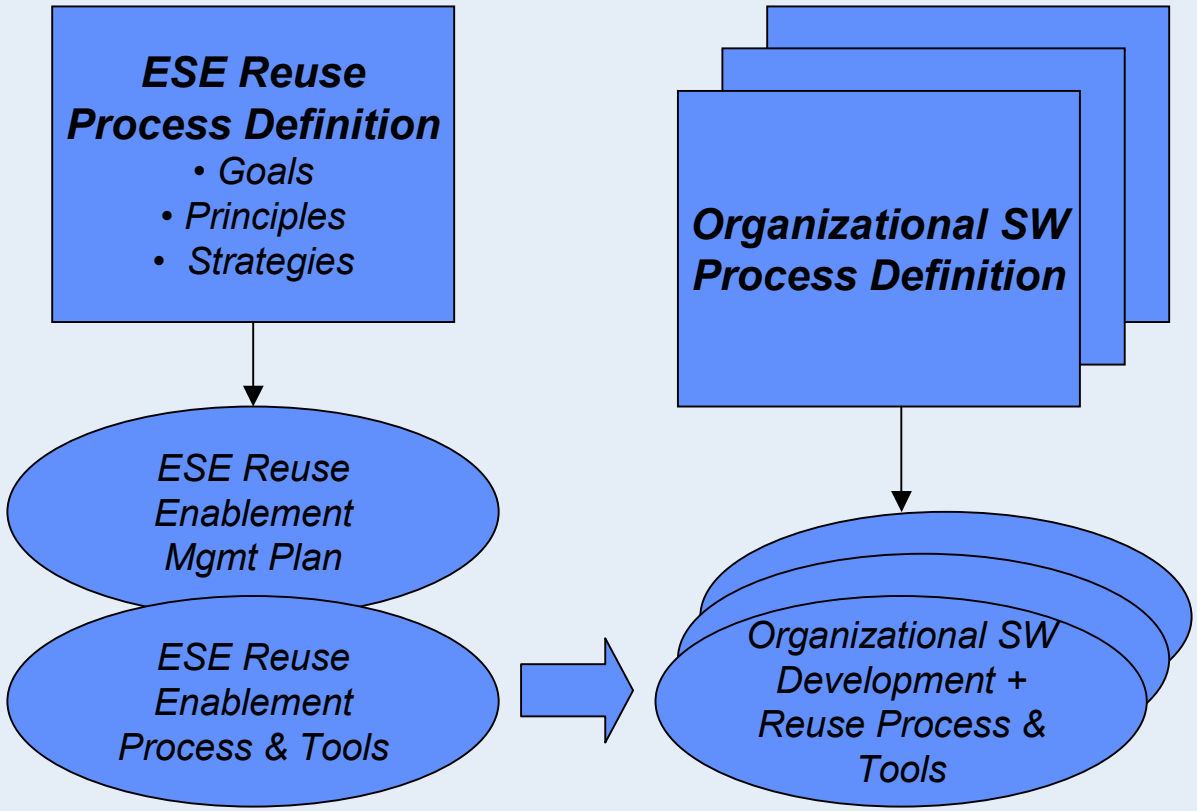
  Focus topics for today's workshop (Guiding principles, Reuse program strategies, Reuse enablement strategies, Evolutionary processes)
- **Wrap-up: key recommendations (15 min)**

Workshop Results Summary

- **Good community participation…thank you!**
  - 10+ active community participants
- **Identified many important aspects of a reuse initiative**
  - 35 problems and opportunities affecting reuse
  - 5 additional guiding principles
  - 8 program strategies
  - 5 technical reuse enablement strategies
  - 3 recommendations on evolution of the process
  - 1 recommendation on use of reference architecture
- **Details of inputs and recommendations included in following slides**
  - Results are only lightly summarized

Workshop Goals

- **Develop high-level recommendations for reuse processes to be incorporated into a SEEDS management plan**
- **Focus on cross-organizational enablement**
  - Independent organizations will use their own processes



Workshop Format

- **Participatory**
  - Study team is only here to get discussion started and capture ideas
  - Study team will synthesize results and plan follow-up sessions
- **Focus on idea generation and capture**
  - Identify as many good ideas as possible
  - Considerations/possibilities are provided to help get you started
    - Add/delete/change as you see fit
- **Prioritizing recommendations to be done later**
  - Consensus is not required at this point
  - Remember: different environments may have different needs!
  - Numbering of ideas is for reference only and does not indicate importance or priority
  - Critiquing of ideas should be limited
  - Criticizing the originator of an idea is not allowed
- **Two teams**
  - **"Mission critical"**: focus on **"improved clone & own"** reuse approach
    - Driven by launch schedules and a need for daily, highly reliable production or archiving needs (e.g. SIPS, DAACs for standard products and high volume distribution)
  - **"Mission success"**: focus on **"open source"** reuse approach
    - Driven more by need for research in science, applications, or information systems, need to experiment with differing products, approaches, mechanisms and adapt to new understandings (e.g. ESIP-2s, -3s, analysis, etc.)

Reuse Goals

- **These are the ultimate qualitative and quantitative results that are desired from a reuse initiative**
- **Established Goals**
  1. Reduce the cost of providing a given set of capabilities in ESE data systems, and thereby enable ESE data systems to better meet mission/science/application requirements within available budgets
  2. Increase the flexibility and responsiveness of ESE data systems to better meet mission/science/application requirements in a timely and cost effective fashion
  3. Increase effective and accountable community participation to better meet ESE goals
  4. Reduce technical/schedule/cost risks associated with developing ESE data systems

Problems & Opportunities

- **What are the positive and negative factors that contribute to the ability to reuse software and meet the stated goals?**
- **Considerations/Possibilities**
  1. Intellectual property policies restrict sharing of reusable assets
  2. Mission-oriented funding does not encourage development of assets for reuse
  3. Developers of potentially reusable components do not have funding to document and support components for use by others
  4. Access to experts (esp., component authors) reduces the effort and risk associated with reusing a component
  5. "NIH"

Problems & Opportunities - Community Input

- **Mission Critical Environments**
  1. Modularity (+)
  2. Highly skilled workforce (+)
  3. Contractual relationships w/ vendors that limit access to software assets (-)
  4. Organizational bias to use certain software (-)
  5. Organizational knowledge about available software (+)
  6. Fit with application (+/-)
  7. Lack of contributions back to asset base (-)
  8. Lack of schedule/funds to make contributions to the asset base (-)
  9. Choice of language, esp. at beginning of mission may or may not current preference (+/-)
  10. Changes to languages/technology (-)
  11. Proven code that does a hard or expensive function (+)
  12. Test cases (+)
  13. Lack of time at the beginning of a mission or on first of several missions (not practical to redevelop after the fact) (-)
  14. No community infrastructure for reuse (e.g., asset library moderator) (-)
  15. Ability to start a small/simple process…get it out there! (+)
  16. Recognition that reuse could shorten schedule or get preliminary functions/foundation up quickly (+)
  17. No existing library of usable assets (-)
  18. NASA culture/mission (technical and management) encourages new tech prototypes over reuse (-)

Problems & Opportunities - Community Input

SELDS FOCUS AREA

- **Mission Success Environments**
  1. Ability to assume some risk in development (+)
  2. Knowledge of--and communication with--those who have software assets (+)
  3. Hesitation to share because code was not designed/documented to be shared (e.g., TRMM IDL viewer) (-)
  4. Modularity (+)
  5. Lack of incentive to reuse (e.g., no payment for support, mission-oriented funding, etc.) (-)
  6. Open source approach…inherently removes barriers, encourages design for reuse, improves quality, incorporates enhancements, results in feeling of shared ownership, etc. (+++)
  7. Licensing that requires contributions back to code base (+)
  8. Reuse approach established within a development team (+)
  9. Acceptance of limitations in existing SW (same as COTS) (+)
  10. Starting as open source vs. releasing to open source through lengthy export review (+)
  11. Issues regarding liability/copyright (-)
  12. Perceptions about security of open source- easy to find security holes (good if the good guys find them, bad if the bad guys find them) (+/-)
  13. Pride over authorship, control over functionality, and ability to match specific needs (-).
  14. Fear of having to support software released for reuse (-)
  15. Concern that effort to make it reusable will be wasted…asset might not be reused (-)
  16. Documentation not in consistent form (-)
  17. No ESE journal that supports advertising available assets (-)
  18. Perceived contribution toward interoperability (+)

Guiding Principles

- **What are the desired high-level attributes of a reuse initiative (and associated processes)?**
- **Considerations/Possibilities**
  1. Defined by actual stakeholders in the ESE community
  2. Tailored to different environments rather than one-size-fits-all
  3. Starts simple and evolves
  4. Ensures a practical focus
  5. Leverages existing activities & organizations
  6. Enables but does not force reuse
  7. Has cross-organizational emphasis…does not dictate individual organization practices
  8. Provides recommendations and guidance, but is not prescriptive

Guiding Principles - Community Input

- **Mission Critical Environments**
  1. Training and knowledge accompanies software asset
  2. Different levels (granularity) of reuse can be appropriate
  3. New code developed should be written with reuse in mind
  4. Use competition to drive reuse
  5. Incentives to enable #3 above (across missions)

Guiding Principles - Community Input

- **Mission Success Environments**
  1. Encourage reuse to be built into the original programming (e.g., team-programming approach, extreme programming)
     - If you're programming it right the first time, then sharing it is not a problem
     - Everyone has ownership of the code
  2. Define criteria to determine appropriateness of reuse
     - Reuse is not always appropriate: encourage reuse only in appropriate areas/components
  3. Focus on areas that are likely to be successful
     - Feedback from community is needed to identify these areas
  4. Determine criteria for components to be included in open source repository
     - Criteria can include community, size of community, interest of other communities in component (likelihood of reuse), structure of software
  5. Create open source authoring environment / infrastructure
  6. Determine the authority(ies) to modify open source software
  7. Include a cookbook for communities to be able to evaluate whether they should provide their software as open source
  8. Be scalable
  9. Define fast and streamlined approval process for proposed open source components
  10. Define peer-reviewed process for selecting components for open source

Reuse Program Strategies

SEEDS Formulation

- **What are the high-level plans of action for a reuse initiative that could be incorporated into a SEEDS program management plan?**
- **Considerations/Possibilities**
  1. Establish two working groups (mission, science/apps) chartered with defining reuse processes and supporting architecture processes
  2. Establish a working group to develop program policies regarding reuse
  3. Establish teams to mine assets (how many?)
  4. Establish forums to share reuse practices
  5. Enlist software engineering specialist support (e.g, CMU Software Engineering Institute)
  6. Enlist stakeholder representatives (or proxies) into the formulation study team until working groups can be established
  7. Top down vs. bottom-up: Start w/ architecture and work down to components vs. start with high-payoff functions and work up to system level
  8. In-place vs. external: have authoring organizations provide reuse support vs. have one or more non-mission orgs support reuse efforts
- **Check: does each program strategy follow the preceding principles?**

Reuse Program Strategies - Community Input

- **Mission Critical Environments**
  1. Start with single working group for mission critical reuse
  2. Use working group(s) to set the policies/targets for level of reuse
  3. Minimize effort expended on infrastructure to enable reuse
  4. Clearly identify originator of software
  5. Have authoring organization provide technical support for reused components (for efficiency)
  6. Provide additional funding to cover overhead for initial activities needed to advertise and package reusable code (this is not to be confused with re-writing for reuse)
  7. Facilitate communication needed for reuse
  8. Do not enlist software engineering specialist support (CMU is overkill)
  9. Develop proper incentives to facilitate reuse

Reuse Program Strategies - Community Input

- **Mission Success Environments**
  1. Create environment for (conformance) testing
  2. Validate actual need for components
  3. Provide institutional support: create a program to encourage reuse
     - includes computers and staff
     - solicits software from community
     - polls community for validation of components
     - makes decisions about whether to support component
     - designs testbeds for testing component
     - identifies champion to oversee and support code
     - publish what's available
  4. Start with prototypes to test/show/compare success
     - Leverage existing reuse-related resources (eg. Source Forge, Google) (issue with licensing)
     - Start with something small, simple and general enough (for interest) and see if it works
       - More general component is likely to draw broader reuse interest
     - Try two prototypes: one involving outside communities and one involving ESE communities only
  5. Do #4 first then #3
  6. Reach out to outside communities that may have interest only in specific components
     - Outside communities (in different domains) may share the need for specific components used in the ESE domain

Reuse Enablement Strategies

- **What specific plans of action and technical approaches should be used to accomplish the stated goals of a software reuse initiative?**
- **Considerations/Possibilities**
  1. Establish a reusable component library
  2. Establish a testbed to help identify and qualify reusable components
  3. Empower a team of experts to evangelize reuse
  4. Develop a software experience library with links to experts, assets, and other resources
  5. Develop a reference architecture to enable component reuse by enhancing component compatibility
  6. Document architecture/design patterns to enable design reuse
  7. Establish policies and incentives that counteract disincentives to reuse (e.g., NIH)
  8. Provides tools (e.g., reuse library software) to support reuse activities
  9. Incorporate reuse into NASA development standards
  10. Use formal methods and application generators to assemble systems from existing assets
- **Check: is each enablement strategy supported by a program strategy?**

Reuse Enablement Strategies - Community Input

- **Mission Critical Environments**
  1. Develop software experience library with links to experts, assets, and other resources
  2. Do NOT focus on component library…#1 above is preferable
  3. Provide standard "checklist" of items that must be provided to make assets reusable; refine the checklist periodically
  4. Do NOT establish a formal independent testbed…"testbedding" should be done at provider and customer site
  5. Conduct reuse workshops/forums to share lessons learned, advertise, obtain ideas, etc. at international meetings and other venues
  6. Document architecture/design patterns to enable design reuse
  7. Do NOT use formal methods and application generators to assemble systems from existing assets…this approach is overkill

Reuse Enablement Strategies - Community Input

- **Mission Success Environments**
  1. Host NASA open source workshops for awareness and training
  2. Collect and publicize success stories
  3. Clarify intellectual property issues
  4. Define management roles and responsibilities to encourage reuse (e.g., encourage team members to participate in open source forums)
  5. Provide incentives for sharing and reuse
  6. Identify and fund software to enable it to become successful open source (e.g., improving quality, going through the steps to provide it as open source, active engagement with community, moderating, etc)
  7. Examine other institutional models (HP, IBM, other governmental organizations, universities)

Evolutionary Processes

- **In what ways can the process evolve over time?**
- **Considerations/Possibilities**
  1. Strategy employed: enhancing communications → in-place-reuse → component library
  2. Specificity: Notional architecture → drill down to concrete architecture
  3. Formality: Prototype process → actual process
  4. Functional scope: highest-payoff functional areas/components → next highest areas

Evolutionary Processes – Community Input

- **In what ways can the process evolve over time?**
- **Mission Critical Environments**
  1. Highly used module could over time be open-sourced
  2. End user feedback to original provider (enhancements, bug fixes, etc.) acts like mini tech infusion, obsolescence fighter
  3. On-going participation of working group: more instances of reuse over time will help to map out the future directions
- **Mission Success Environments**
  1. Start with prototype process and learn from experience
  2. Identify and extend existing tools to support community needs (e.g., extend ENVI to support HDF)
  3. Start with high payoff and highly reusable areas

Reference Architecture Purpose

- **How should a reference architecture be used to enhance software reuse?**
- **Considerations/Possibilities**
  1. Enhance communications between groups with software assets to share
  2. Standardize definition/functionality of components to help categorize components, enhance compatibility, and enable components to be independently acquired
  3. Make near-plug-and-play integration of components possible
  4. Enable black-box plug & play

Reference Architecture Purpose - Community Input

- **Mission Critical Environments**
  1. Enhance communications between groups with software assets to share
- **Mission Success Environments**
  1. Enhance communications between groups with software assets to share
  2. Standardize definition/functionality of components to help categorize components, enhance compatibility, and enable components to be independently acquired
  3. As a guide, communication mechanism
  4. Document set of requirements that a data system would meet (components needed, etc.)
  5. Enable reusing requirement specs
  6. Document criteria for accepting components
  7. Create a market for components

Reuse Initiative Models

- **What existing organizations and processes should NASA's ESE use as a model for implementing a reuse effort?**
- **Considerations/Possibilities**
  1. ESIP Federation IDL cluster (plug-in components)
  2. MODAPS/SeaWIFS/DODS (various strategies)
  3. Earth Science Modeling Framework (common tools, architecture/conventions)
  4. National HPCC Software Exchange (reuse tool broker)
  5. Reuse Information Clearinghouse (information dissemination)
  6. National Association of State Chief Information Officers ComponentSource (library)
  7. Workshop on Institutionalizing Software Reuse (information dissemination)
- **Check: do the program strategies cover the key strategies used by these models?**

Note: there was not sufficient time to address this topic during the breakout sessions

Wrap-Up: Community Input Highlights

- **Mission Critical Environments**
  - Don't spend a lot of effort creating a reuse infrastructure
    - Focus efforts on actual reuse efforts, not preparing for reuse
  - Provide appropriate incentives to encourage reuse
- **Mission Success Environments**
  - Don't spend a lot of effort creating a reuse infrastructure
    - Use existing open source infrastructure (e.g., Freshmeat.net)
  - Start with a simple process and evolve